



**Microsoft**

# 70-505-CSharp

*TS- Microsoft .NET Framework 3.5 Windows Forms Application Development*

controls. Set the ColumnCount property to 4. Then add the nested controls to the RowCount property.

H. Create a FlowLayoutPanel control to contain the controls, and add the nested controls to it. Call the ResumeLayout() method with the fourth nested control and every fourth control after that.

**Answer:** A

**QUESTION:** 98

You are developing a .NET Framework 3.5 Windows Forms application. The application will download a 300 MB audio file from a Windows Communication Foundation (WCF) Web service by using the BackgroundWorker class.

You need to ascertain when the BackgroundWorker has finished the file download. What should you do?

- A. Register for the ProgressChanged event.
- B. Call the GetService() method.
- C. Get the IsBusy property value.
- D. Call the RunWorkerAsync() method.
- E. Get the WorkerSupportsCancellation property value.
- F. Call the OnRunWorkerComplete() method.
- G. Register for the DoWork event.
- H. Get the WorkerReportsProgress property value.

**Answer:** A

**QUESTION:** 99

You create Windows Forms applications by using the .NET Framework 3.5. You plan to use the Windows Installer to deploy a new application.

The application must meet the following requirements:

- Support deployment to 32-bit and 64-bit operating systems.
- Use the 64-bit Program Files folder when deployed to 64-bit platforms.

You need to ensure that the application is deployed appropriately.

What should you do?

- A. • Create an MSI file that is targeted to 64-bit platforms.  
• Create an MSI file that is targeted to 32-bit platforms.
- B. • Create a single MSI file.  
• Create a merge module that contains the 32-bit and 64-bit code.
- C. • Create a single MSI file.

- Add a launch condition that is set to Version NT64.
- D. • Create a single MSI file.
- Add a launch condition that is set to NOT Version NT64.

**Answer:** A

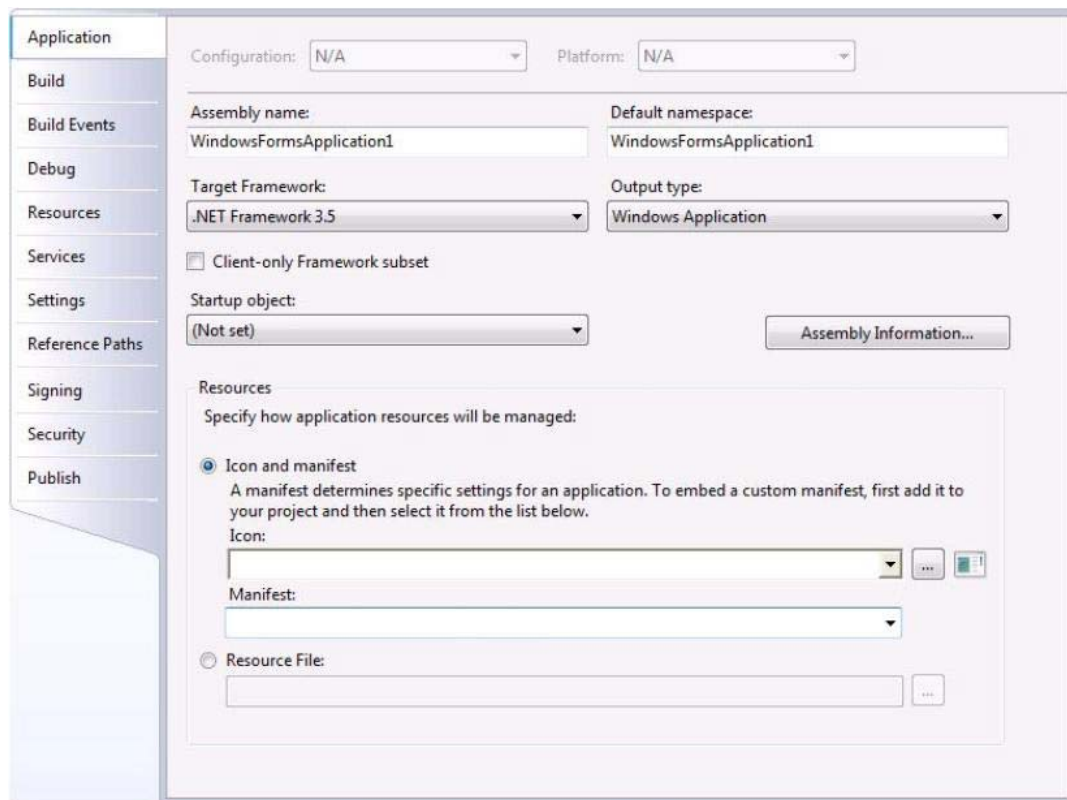
## **QUESTION:** 100

### **HOTSPOT**

You are developing a Microsoft Visual Studio Tools for Office (VSTO) application by using the .NET Framework 3.5. Users will download and install the application through a ClickOnce deployment. You need to select the correct manifest setting to manage the application resources.

Which manifest setting should you use?

To answer, select the appropriate setting in the answer area.



The screenshot shows the 'Application' settings window in Visual Studio. The left sidebar has tabs for Application, Build, Build Events, Debug, Resources, Services, Settings, Reference Paths, Signing, Security, and Publish. The 'Resources' tab is selected. The main area shows configuration for the application. At the top, 'Configuration' and 'Platform' are both set to 'N/A'. Below, 'Assembly name' is 'WindowsFormsApplication1' and 'Default namespace' is 'WindowsFormsApplication1'. 'Target Framework' is '.NET Framework 3.5' and 'Output type' is 'Windows Application'. There is a checkbox for 'Client-only Framework subset' which is unchecked. 'Startup object' is '(Not set)'. There is an 'Assembly Information...' button. The 'Resources' section is expanded, showing 'Specify how application resources will be managed:'. There are two radio buttons: 'Icon and manifest' (which is selected) and 'Resource File'. Below 'Icon and manifest', there is a description: 'A manifest determines specific settings for an application. To embed a custom manifest, first add it to your project and then select it from the list below.' There are two dropdowns: 'Icon:' and 'Manifest:'. The 'Manifest:' dropdown is currently empty. Below 'Resource File', there is a text box and a button.

Application

Build

Build Events

Debug

Resources

Services

Settings

Reference Paths

Signing

Security

Publish

Configuration: N/A Platform: N/A

Assembly name: WindowsFormsApplication1 Default namespace: WindowsFormsApplication1

Target Framework: .NET Framework 3.5 Output type: Windows Application

☐ Client-only Framework subset

Startup object: (Not set) Assembly Information...

Resources

Specify how application resources will be managed:

☒ Icon and manifest

A manifest determines specific settings for an application. To embed a custom manifest, first add it to your project and then select it from the list below.

Icon:

Embed manifest with default settings  
Create application without a manifest  
Properties\app.manifest

☐ Resource File:

**Answer:**

Application
Build
Build Events
Debug
Resources
Services
Settings
Reference Paths
Signing
Security
Publish

Configuration: N/A
Platform: N/A

Assembly name:
WindowsFormsApplication1

Default namespace:
WindowsFormsApplication1

Target Framework:
.NET Framework 3.5

Output type:
Windows Application

☐ Client-only Framework subset

Startup object:
(Not set)

Assembly Information...

Resources
Specify how application resources will be managed:

☒ Icon and manifest

A manifest determines specific settings for an application. To embed a custom manifest, first add it to your project and then select it from the list below.

Icon:

Embed manifest with default settings
Create application without a manifest
Properties\app.manifest

☐ Resource File:

## Explanation:

Application
Build
Build Events
Debug
Resources
Services
Settings
Reference Paths
Signing
Security
Publish

Configuration: N/A
Platform: N/A

Assembly name:
WindowsFormsApplication1

Default namespace:
WindowsFormsApplication1

Target Framework:
.NET Framework 3.5

Output type:
Windows Application

☐ Client-only Framework subset

Startup object:
(Not set)

Assembly Information...

Resources
Specify how application resources will be managed:

☒ Icon and manifest

A manifest determines specific settings for an application. To embed a custom manifest, first add it to your project and then select it from the list below.

Icon:

Embed manifest with default settings
Create application without a manifest
Properties\app.manifest

☐ Resource File:

82

C:\Users\Kamran\Desktop\image.JPG

**QUESTION:** 101

**DRAG DROP**

You are creating a .NET Framework 3.5 Windows Forms application.

The application will display and allow changes of telephone numbers through a control class named PhoneTextBox. A form named ContactWindow will use the PhoneTextBox control. You need to create the PhoneTextBox control and then add an instance of the PhoneTextBox control to the ContactWindow form.

Which actions should you perform in sequence?

To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.

Build the Solution.	
Drag <b>TextBox</b> controls for each component of the address from the Toolbox to the designer.	
Use the <b>Add New Item</b> command to add a User Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.	
Use the <b>Add New Item</b> command to add a Custom Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.	
In the code for PhoneTextBox.cs, declare a delegate in the <b>PhoneTextBox</b> class for the <b>Text</b> property of each <b>TextBox</b> object.	
Open ContactWindow.cs in the designer. Drag a <b>PhoneTextBox</b> control from the Toolbox to the designer. Set the properties of the <b>PhoneTextBox</b> control.	
In the code for ContactWindow.cs, add code to the constructor of the <b>ContactWindow</b> class to create a <b>PhoneTextBox</b> control and then set its properties.	
In the code for PhoneTextBox.cs, add a property to the <b>PhoneTextBox</b> class for each of the <b>TextBox</b> objects to allow the <b>Text</b> property of the <b>TextBox</b> object to be accessed and/or modified.	

**Answer:**

Build the Solution.	Build the Solution.
Drag <b>TextBox</b> controls for each component of the address from the Toolbox to the designer.	
Use the <b>Add New Item</b> command to add a User Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.	Open ContactWindow.cs in the designer. Drag a <b>PhoneTextBox</b> control from the Toolbox to the designer. Set the properties of the <b>PhoneTextBox</b> control.
Use the <b>Add New Item</b> command to add a Custom Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.	In the code for PhoneTextBox.cs, add a property to the <b>PhoneTextBox</b> class for each of the <b>TextBox</b> objects to allow the <b>Text</b> property of the <b>TextBox</b> object to be accessed and/or modified.
In the code for PhoneTextBox.cs, declare a delegate in the <b>PhoneTextBox</b> class for the <b>Text</b> property of each <b>TextBox</b> object.	Drag <b>TextBox</b> controls for each component of the address from the Toolbox to the designer.
Open ContactWindow.cs in the designer. Drag a <b>PhoneTextBox</b> control from the Toolbox to the designer. Set the properties of the <b>PhoneTextBox</b> control.	Use the <b>Add New Item</b> command to add a Custom Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.
In the code for ContactWindow.cs, add code to the constructor of the <b>ContactWindow</b> class to create a <b>PhoneTextBox</b> control and then set its properties.	
In the code for PhoneTextBox.cs, add a property to the <b>PhoneTextBox</b> class for each of the <b>TextBox</b> objects to allow the <b>Text</b> property of the <b>TextBox</b> object to be accessed and/or modified.	

### Explanation:

	Build the Solution.
	Open ContactWindow.cs in the designer. Drag a <b>PhoneTextBox</b> control from the Toolbox to the designer. Set the properties of the <b>PhoneTextBox</b> control.
	In the code for PhoneTextBox.cs, add a property to the <b>PhoneTextBox</b> class for each of the <b>TextBox</b> objects to allow the <b>Text</b> property of the <b>TextBox</b> object to be accessed and/or modified.
Use the <b>Add New Item</b> command to add a Custom Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.	Drag <b>TextBox</b> controls for each component of the address from the Toolbox to the designer.
In the code for PhoneTextBox.cs, declare a delegate in the <b>PhoneTextBox</b> class for the <b>Text</b> property of each <b>TextBox</b> object.	Use the <b>Add New Item</b> command to add a User Control named PhoneTextBox to the project. Open PhoneTextBox.cs in the Windows Forms Designer.
In the code for ContactWindow.cs, add code to the constructor of the <b>ContactWindow</b> class to create a <b>PhoneTextBox</b> control and then set its properties.	

C:\Users\Kamran\Desktop\image.JPG

### QUESTION: 102

#### DRAG DROP

You are creating a .NET Framework 3.5 Windows Forms application.

A control class will change its foreground color as users enter text into the control or selects a color from the control list. You need to create the control class so that the



foreground color of the control changes to yellow when the user enters the letters "yellow" or selects yellow from the control list. How should you complete the code segment?

To answer, drag the appropriate method names or class names to the correct location or locations in the answer area.

Answer Choices	Answer Area
<input type="text" value="OnKeyUp"/>	<pre> public class ColoredTextBox :  {     protected override void (System.Windows.Forms.KeyEventArgs e)     {         if (this.SelectedText.ToLower() == "yellow")             this.Foreground = System.Drawing.Color.Yellow;         else             this.Foreground = System.Drawing.Color.Black;         base. (e);     }     protected override void (EventArgs e)     {         if (this.SelectedText.ToLower() == "yellow")             this.Foreground = System.Drawing.Color.Yellow;         else             this.Foreground = System.Drawing.Color.Black;         base. (e);     } } </pre>
<input type="text" value="OnKeyDown"/>	
<input type="text" value="OnTextChanged"/>	
<input type="text" value="OnVisibleChanged"/>	
<input type="text" value="OnForeColorChanged"/>	
<input type="text" value="OnBackColorChanged"/>	
<input type="text" value="System.Windows.Forms.ComboBox"/>	
<input type="text" value="System.Web.UI.WebControls.ComboBox"/>	
<input type="text" value=""/>	
<input type="text" value=""/>	

**Answer:**

Answer Choices	Answer Area
<input type="text" value="OnKeyUp"/>	<pre> public class ColoredTextBox : System.Windows.Forms.ComboBox {     protected override void OnKeyUp (System.Windows.Forms.KeyEventArgs e)     {         if (this.SelectedText.ToLower() == "yellow")             this.Foreground = System.Drawing.Color.Yellow;         else             this.Foreground = System.Drawing.Color.Black;         base.OnKeyUp (e);     }     protected override void OnTextChanged (EventArgs e)     {         if (this.SelectedText.ToLower() == "yellow")             this.Foreground = System.Drawing.Color.Yellow;         else             this.Foreground = System.Drawing.Color.Black;         base.OnTextChanged (e);     } } </pre>
<input type="text" value="OnKeyDown"/>	
<input type="text" value="OnTextChanged"/>	
<input type="text" value="OnVisibleChanged"/>	
<input type="text" value="OnForeColorChanged"/>	
<input type="text" value="OnBackColorChanged"/>	
<input type="text" value="System.Windows.Forms.ComboBox"/>	
<input type="text" value="System.Web.UI.WebControls.ComboBox"/>	
<input type="text" value=""/>	
<input type="text" value=""/>	

**Explanation:**

Answer Choices	Answer Area
<input type="text" value="OnKeyUp"/>	<pre> public class ColoredTextBox : System.Windows.Forms.ComboBox {     protected override void OnKeyUp (System.Windows.Forms.KeyEventArgs e)     {         if (this.SelectedText.ToLower() == "yellow")             this.Foreground = System.Drawing.Color.Yellow;         else             this.Foreground = System.Drawing.Color.Black;         base.OnKeyUp (e);     }     protected override void OnTextChanged (EventArgs e)     {         if (this.SelectedText.ToLower() == "yellow")             this.Foreground = System.Drawing.Color.Yellow;         else             this.Foreground = System.Drawing.Color.Black;         base.OnTextChanged (e);     } } </pre>
<input type="text" value="OnKeyDown"/>	
<input type="text" value="OnTextChanged"/>	
<input type="text" value="OnVisibleChanged"/>	
<input type="text" value="OnForeColorChanged"/>	
<input type="text" value="OnBackColorChanged"/>	
<input type="text" value="System.Windows.Forms.ComboBox"/>	
<input type="text" value="System.Web.UI.WebControls.ComboBox"/>	
<input type="text" value=""/>	
<input type="text" value=""/>	

C:\Users\Kamran\Desktop\image.JPG



**QUESTION:** 103

**DRAG DROP**

You are developing a .NET Framework 3.5 Windows Forms application. You create a custom control on a form by creating a new class. You declare the class by using the following code segment:

```
public class MyCustomControl:Control{ }
```

You use the OnPaint() method to render the control. When you run the application, the control does not render on the form.

You need to ensure that the control can render by using the OnPaint() method.

How should you implement the OnPaint() method?

To answer, drag the appropriate code segment or segments to the correct location or locations in the work area.

Answer Choices	Answer Area
<input type="text" value="bool"/>	<code>protected</code> <input type="text" value=""/> <code>OnPaint(</code> <input type="text" value=""/> <code>e)</code>
<input type="text" value="override bool"/>	<code>{ }</code>
<input type="text" value="void"/>	
<input type="text" value="override void"/>	
<input type="text" value="DrawEventArgs"/>	
<input type="text" value="OnDrawEventArgs"/>	
<input type="text" value="PaintEventArgs"/>	
<input type="text" value="OnPaintEventArgs"/>	

**Answer:**

Answer Choices	Answer Area
<input type="text" value="bool"/>	<code>protected</code> <input type="text" value="override void"/> <code>OnPaint(</code> <input type="text" value="OnDrawEventArgs"/> <code>e)</code>
<input type="text" value="override bool"/>	<code>{ }</code>
<input type="text" value="void"/>	
<input type="text" value="override void"/>	
<input type="text" value="DrawEventArgs"/>	
<input type="text" value="OnDrawEventArgs"/>	
<input type="text" value="PaintEventArgs"/>	
<input type="text" value="OnPaintEventArgs"/>	

**Explanation:**

Answer Choices	Answer Area
<input type="text" value="bool"/>	<pre>protected override void OnPaint(<input type="text" value="PaintEventArgs"/> e) { }</pre>
<input type="text" value="override bool"/>	
<input type="text" value="void"/>	
<input type="text" value="DrawEventArgs"/>	
<input type="text" value="OnDrawEventArgs"/>	
<input type="text" value=""/>	
<input type="text" value="OnPaintEventArgs"/>	

C:\Users\Kamran\Desktop\image.JPG

Download Full Version From <https://www.certkillers.net>



**DON'T KNOW**  
OR NO PREFERENCE

*Pass your exam at First Attempt....Guaranteed!*